

# LAB#5

---

Sorting

---

# Bubble sort

---

## Bubble sort :

- **Bubble sort** is a simple sorting algorithm. The algorithm starts at the beginning of the data set.
    - compares the first two elements.
    - If the first is greater than the second, swaps them.
    - continues doing this for each pair of adjacent elements
    - Starts again with the first two elements, repeating until no swaps have occurred on the last pass.
-

# Bubble sort

---

```
#include <iostream>
using namespace std;
int array1[10]={20,3,12,4,65,23,10,14,6,19};
void display();
void swap(int & , int &);

void main()
{
int size=10;
display();
for(int out = size-1; out>1; out--)
    {for(int in=0;in<out;in++)
        {if(array1[in] > array1[in + 1])
            swap(array1[in], array1[in + 1]);}
        display();}
}
```

---

# Bubble sort

---

```
void swap (int &x, int &y)  
{  
int temp=x;  
x=y;  
y=temp;  
}
```

```
void display()  
{  
for(int i=0;i<10;i++)  
cout << array1[i] << " ";  
cout << "\n" << "-----\n";  
}
```

---

```
C:\Windows\system32\cmd.exe

20 3 12 4 65 23 10 14 6 19
-----
3 12 4 20 23 10 14 6 19 65
-----
3 4 12 20 10 14 6 19 23 65
-----
3 4 12 10 14 6 19 20 23 65
-----
3 4 10 12 6 14 19 20 23 65
-----
3 4 10 6 12 14 19 20 23 65
-----
3 4 6 10 12 14 19 20 23 65
-----
3 4 6 10 12 14 19 20 23 65
-----
Press any key to continue . . .
```

# Bubble sort

6 5 3 1 8 7 2 4

# Selection sort

---

## Selection sort :

- **Selection sort**

1. Find the minimum value in the list
2. Swap it with the value in the first position
3. Repeat the steps above for the remainder of the list (starting at the second position and advancing each time)

# Selection sort

---

```
#include <iostream>
using namespace std;
int array1[10]={20,3,12,4,65,23,10,14,6,19};
int size=10;
void display();
int minimum(int);
void swap(int & , int &);

void main()
{
display();
for(int i=0; i < size-1; i++)
{
    swap(array1[i], array1[minimum(i)]);
    display();
}
}
```

---

# Selection sort

```
int minimum(int pos)
{
    int min_val=array1[pos];
    int min_pos=pos;
    for(int k=pos+1; k<size; k++)
    { if(array1[k] < min_val)
        {min_val=array1[k];
        min_pos=k; }
    }
    return min_pos; }

void swap (int &x, int &y)
{ int temp=x;
  x=y;
  y=temp; }

void display()
{for(int i=0;i<10;i++)
  cout << array1[i] << " ";
  cout << "\n" << "-----\n"; }
```

A screenshot of a Windows command prompt window. The title bar shows the path 'C:\Windows\system32\cmd.exe'. The window contains a 10x10 grid of numbers, with each row separated by a dashed line. The numbers are as follows:

20	3	12	4	65	23	10	14	6	19
3	20	12	4	65	23	10	14	6	19
3	4	12	20	65	23	10	14	6	19
3	4	6	20	65	23	10	14	12	19
3	4	6	10	65	23	20	14	12	19
3	4	6	10	12	23	20	14	65	19
3	4	6	10	12	14	20	23	65	19
3	4	6	10	12	14	19	23	65	20
3	4	6	10	12	14	19	20	65	23
3	4	6	10	12	14	19	20	23	65

Press any key to continue . . .

5 1 12 -5 16 2 12 14

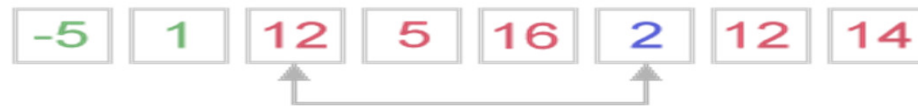
5 1 12 -5 16 2 12 14



-5 1 12 5 16 2 12 14



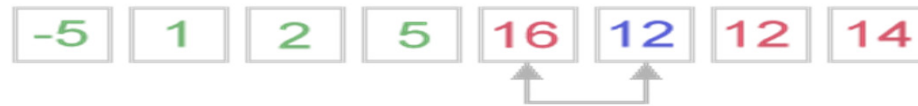
-5 1 12 5 16 2 12 14



-5 1 2 5 16 12 12 14



-5 1 2 5 16 12 12 14



-5 1 2 5 12 16 12 14



-5 1 2 5 12 12 16 14



-5 1 2 5 12 12 14 16

# Insertion sort

---

## Insertion sort :

- **Insertion sort** is substantially better than the bubble sort.
    - In the **outer** for loop, out starts at 1 and moves right. It marks the leftmost unsorted data.
    - In the **inner** while loop, in starts at out and moves left, until either temp is smaller than the array element there, or it can't go left any further.
    - Each pass through the while loop shifts another sorted element one space right.
-

# Insertion sort

---

```
#include <iostream>
using namespace std;
int array1[10]={20,3,12,4,65,23,10,14,6,19};
int size=10;
int temp;
void display();

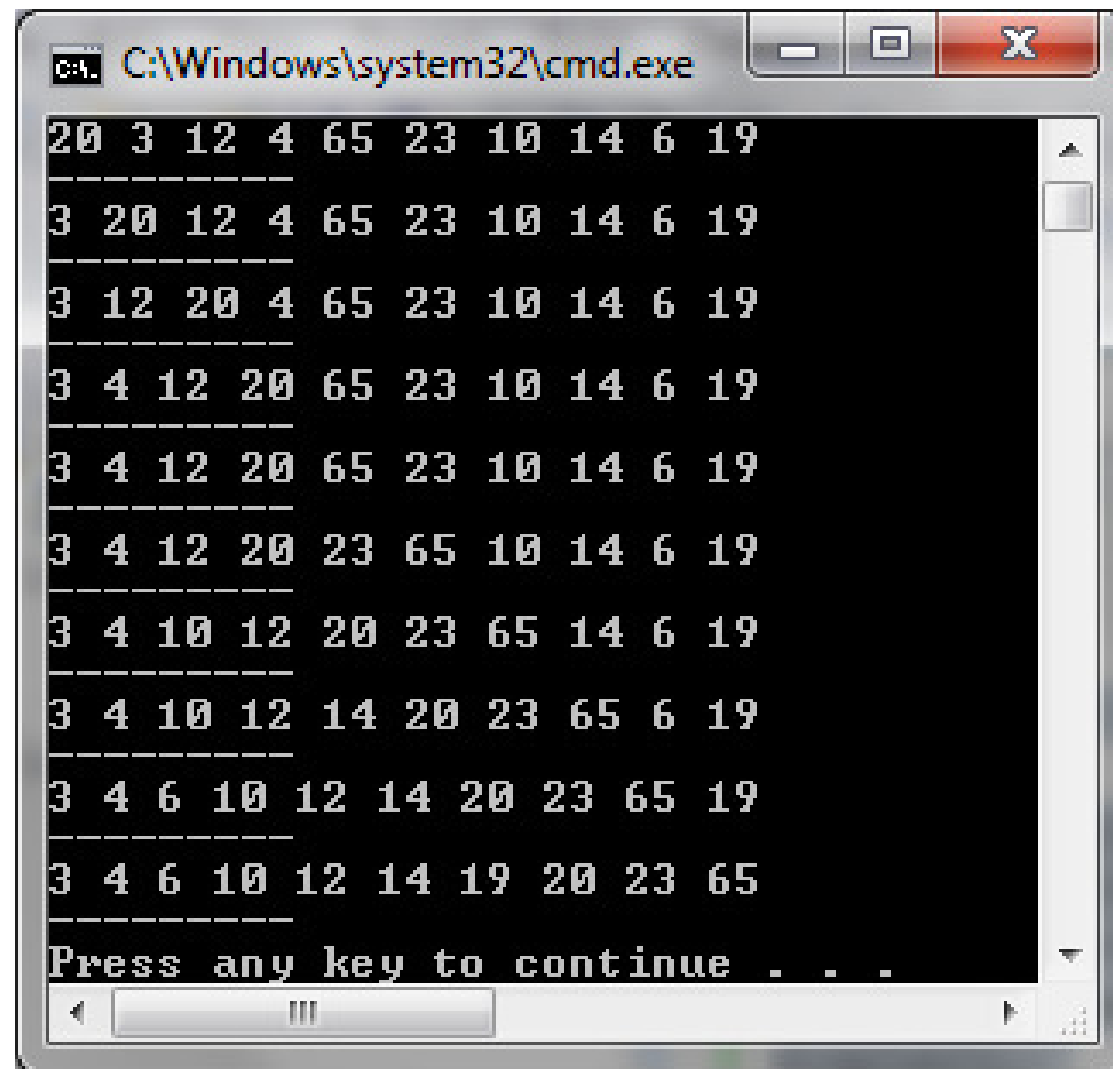
void main() {
display();
int j;
    for(int i=1; i < size; i++) {
        temp=array1[i];
        for(j = i; j> 0 && temp < array1[j-1]; j--)
            array1[j]=array1[j-1];
        array1[j]=temp;
        display();}
}
```

---

# Insertion sort

---

```
void display()  
{  
for(int i=0;i<10;i++)  
cout << array1[i] << " ";  
cout << "\n" << "-----\n";  
}
```



```
C:\Windows\system32\cmd.exe

20 3 12 4 65 23 10 14 6 19
-----
3 20 12 4 65 23 10 14 6 19
-----
3 12 20 4 65 23 10 14 6 19
-----
3 4 12 20 65 23 10 14 6 19
-----
3 4 12 20 65 23 10 14 6 19
-----
3 4 12 20 23 65 10 14 6 19
-----
3 4 10 12 20 23 65 14 6 19
-----
3 4 10 12 14 20 23 65 6 19
-----
3 4 6 10 12 14 20 23 65 19
-----
3 4 6 10 12 14 19 20 23 65
-----
Press any key to continue . . .
```

# Insertion sort

6 5 3 1 8 7 2 4